

"Express Mail" mailing label number: EL513142995USDate of Deposit: November 28, 2001

FORM PTO-1390 (REV. 5-93)		U.S. DEPARTMENT OF COMMERCE PATENT AND TRADEMARK OFFICE		CASE NO. 56/362	
TRANSMITTAL LETTER TO THE UNITED STATES DESIGNATED/ELECTED OFFICE (DO/EO/US) CONCERNING A FILING UNDER 35 U.S.C. 371				U.S. APPLICATION NO. (If known, see 37 C.F.R. 1.5) 09/980303	
INTERNATIONAL APPLICATION NO. PCT/EP 00/04856		INTERNATIONAL FILING DATE May 27, 2000		PRIORITY DATE CLAIMED May 28, 1999	
TITLE OF INVENTION METHOD FOR THE SYNCHRONIZED START-UP OF A NUMERICAL CONTROL					
APPLICANT(S) FOR DO/EO/US Christian Rutkowski and Thomas Klughammer					
Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:					
1. <input checked="" type="checkbox"/> This is a FIRST submission of items concerning a filing under 35 U.S.C. 371					
2. <input type="checkbox"/> This is a SECOND or SUBSEQUENT submission of items concerning a filing under 35 U.S.C. 371					
3. <input checked="" type="checkbox"/> This express request to begin national examination procedures (35 U.S.C. 371(f)) at any time rather than delay examination until the expiration of the applicable time limit set in 35 U.S.C. 371(b) and PCT Articles 22 and 39(1).					
4. <input checked="" type="checkbox"/> A proper Demand for International Preliminary Examination was made by the 19th month from the earliest claimed priority date.					
5. <input checked="" type="checkbox"/> A copy of the International Application as filed (35 U.S.C. 371(c)(2)).					
a. <input checked="" type="checkbox"/> is transmitted herewith (required only if not transmitted by the International Bureau).					
b. <input type="checkbox"/> has been transmitted by the International Bureau.					
c. <input type="checkbox"/> is not required, as the application was filed in the United States Receiving Office (RO/US).					
6. <input checked="" type="checkbox"/> A translation of the International Application into English (35 U.S.C. 371(c)(2)).					
7. <input checked="" type="checkbox"/> Amendments to the claims of the International Application under PCT Article 19 (35 U.S.C. 371(c)(3)).					
a. <input type="checkbox"/> are transmitted herewith (required only if not transmitted by the International Bureau).					
b. <input checked="" type="checkbox"/> have been transmitted by the International Bureau.					
c. <input type="checkbox"/> have not been made; however, the time limit for making such amendments has NOT expired.					
d. <input type="checkbox"/> have not been made and will not be made.					
8. <input checked="" type="checkbox"/> A translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371(c)(3)) (One (1) Amended Sheet for Specification, three (3) Amended Sheets for the Claims).					
9. <input type="checkbox"/> An oath or declaration of the inventor(s) (35 U.S.C. 371(c)(4)).					
10. <input type="checkbox"/> A translation of the annexes to the International Preliminary Examination Report under PCT Article 36 (35 U.S.C. 371(c)(5)) and/or amendments under Article 34.					
Items 11. to 16. Below concern other document(s) or information included:					
11. <input type="checkbox"/> An Information Disclosure Statement under 37 CFR 1.97 and 1.98.					
12. <input type="checkbox"/> An assignment document for recording. A separate cover sheet in compliance with 37 CFR 3.28 and 3.31 is included.					
13. <input checked="" type="checkbox"/> A FIRST preliminary amendment.					
<input type="checkbox"/> A SECOND or SUBSEQUENT preliminary amendment.					
14. <input type="checkbox"/> A substitute specification.					
15. <input type="checkbox"/> A change of power of attorney and/or address letter.					
16. <input type="checkbox"/> Other items or information:					

U.S. APPLICATION NO. (If known, see 37 CFR 1.50) 097980303	INTERNATIONAL APPLICATION NO. PCT/EP 00/04856	CASE NO. 56/362																												
17. <input checked="" type="checkbox"/> The following fees are submitted: Basic National Fee (37 CFR 1.492(a)(1)-(5)): Search Report has been prepared by the EPO or JPO\$890.00 International preliminary examination fee paid to USPTO (37 CFR 1.492(2)(1))\$710.00 No international preliminary examination fee paid to USPTO (37 CFR 1.482) but international search fee paid to USPTO (37 CFR 1.492(a)(2))\$740.00 Neither international preliminary examination fee (37 CFR 1.482) nor international search fee (37 CFR 1.492(a)(3)) paid to USPTO\$1,040.00 International preliminary examination fee paid to USPTO (37 CFR 1.482) and all claims satisfied provisions of PCT Article 33(1)-(4)\$100.00 <div style="text-align: center; border: 1px solid black; padding: 5px; margin: 10px 0;"> ENTER APPROPRIATE BASIC FEE AMOUNT </div>		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th style="width: 50%;">CALCULATIONS</th> <th style="width: 50%;">PTO USE ONLY</th> </tr> <tr> <td style="height: 150px; vertical-align: bottom;">\$890.00</td> <td></td> </tr> </table>		CALCULATIONS	PTO USE ONLY	\$890.00																								
CALCULATIONS	PTO USE ONLY																													
\$890.00																														
Surcharge of \$130.00 for furnishing the oath or declaration later than <input type="checkbox"/> 20 <input checked="" type="checkbox"/> 30 months from the earliest claimed priority date (37 CFR 1.492(e)).		\$130.00																												
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 20%;">Claims</th> <th style="width: 20%;">Number Filed</th> <th style="width: 20%;">Number Extra</th> <th style="width: 20%;">Rate</th> <th style="width: 20%;"></th> </tr> </thead> <tbody> <tr> <td>Total Claims</td> <td>32 - 20 =</td> <td>12</td> <td>x \$ 18.00</td> <td>\$216.00</td> </tr> <tr> <td>Independent Claims</td> <td>2 - 3 =</td> <td>0</td> <td>x \$ 84.00</td> <td>\$0.00</td> </tr> <tr> <td colspan="4">Multiple dependent claim(s) if Applicable)</td> <td>0 + \$280.00</td> </tr> <tr> <td colspan="4" style="text-align: right;">TOTAL OF ABOVE CALCULATIONS =</td> <td>\$1,236.00</td> </tr> </tbody> </table>	Claims	Number Filed	Number Extra	Rate		Total Claims	32 - 20 =	12	x \$ 18.00	\$216.00	Independent Claims	2 - 3 =	0	x \$ 84.00	\$0.00	Multiple dependent claim(s) if Applicable)				0 + \$280.00	TOTAL OF ABOVE CALCULATIONS =				\$1,236.00	Reduction by 1/2 for filing by small entity, if applicable. Small Entity statement must also be filed. (Note 37 CFR 1.9, 1.27, 1.28)			SUBTOTAL = \$1,236.00	
Claims	Number Filed	Number Extra	Rate																											
Total Claims	32 - 20 =	12	x \$ 18.00	\$216.00																										
Independent Claims	2 - 3 =	0	x \$ 84.00	\$0.00																										
Multiple dependent claim(s) if Applicable)				0 + \$280.00																										
TOTAL OF ABOVE CALCULATIONS =				\$1,236.00																										
Surcharge of \$130.00 for furnishing the English translation later than the <input type="checkbox"/> 20 <input type="checkbox"/> 30 months from the earliest claimed priority date (37 CFR 1.492(f)).		\$			TOTAL NATIONAL FEE= \$1,236.00																									
Fee for recording the enclosed assignment (37 CFR 1.21(h)). The assignment must be accompanied by an appropriate cover sheet (37 CFR 3.28, 3.31), \$40.00 per property +		TOTAL FEES ENCLOSED= \$1,236.00			Amount to be refunded \$																									
a. <input checked="" type="checkbox"/> A check in the amount of \$1,236.00 to cover the above fees is enclosed.		charged \$																												
b. <input type="checkbox"/> Please charge my Deposit Account No. 23-1925 in the amount of \$ to cover the above fees. A duplicate copy of this sheet is enclosed.																														
c. <input checked="" type="checkbox"/> The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. 23-1925. A duplicate copy of this sheet is enclosed.																														
NOTE: Where an appropriate time limit under 37 CFR 1.494 or 1.495 has not been met, a petition to revive (37 CFR 1.137(a) or (b)) must be filed and granted to restore the application to pending status.																														
Send All Correspondence to: Brinks Hofer Gilson & Lione P.O. Box 10395 Chicago, IL 60610		Signature <i>John C. Freeman</i> Name John C. Freeman Registration Number 34,483																												



00757

PATENT & TRADEMARK OFFICE

"Express Mail" mailing label number
EL513142995US

Date of Deposit: November 28, 2001

PATENT
CASE NO. 56/362

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application:)
Christian Rutkowski et al.)
Group Art Unit: unassigned)
International Patent Application)
No. PCT/EP00/04856)
International Filing)
Date: May 27, 2000)
U.S. Patent Application)
Serial No.: unassigned)
Filed: November 28, 2001)
Examiner: unassigned)
For: METHOD FOR THE)
SYNCHRONIZED START-UP OF)
A NUMERICAL CONTROL)

PRELIMINARY AMENDMENT

Commissioner for Patents
Washington, D.C. 20231

Dear Sir:

This Preliminary Amendment is being presented to better describe Applicants' claimed invention and it is believed does not present any new matter. Please amend the specification and the claims as follows:

In the Specification:

On page 1, after the title and before line 3, insert the following new paragraphs and headings as follows:

5 Applicants claim, under 35 U.S.C. §§ 120 and 365, the benefit of priority of the filing date of May 27, 2000 of a Patent Cooperation Treaty patent application, copy attached, Serial Number PCT/EP00/04856, filed on the aforementioned date, the entire contents of which are incorporated herein by reference, wherein Patent Cooperation Treaty patent application Serial Number PCT/EP00/04856 was not published under PCT Article 21(2) in English.

10 Applicants claim, under 35 U.S.C. § 119, the benefit of priority of the filing date of May 28, 1999 of a German patent application, copy attached, Serial Number 199 24 461.8, filed on the aforementioned date, the entire contents of which are incorporated herein by reference.

Background of the Invention

Field of the Invention

Replace the paragraph beginning on page 1, line 3 with the following paragraph:

The present invention relates to a method for the synchronized start-up of a numerical control.

20 On page 1, between lines 4 and 5, insert the following heading:

Description of the Related Art

Replace the paragraph beginning on page 1, line 5 with the following paragraph:

EP 0 524 344 B1 discloses a configurable machine tool control having several task-oriented units, for example a numeric and a memory-programmable control element, an operating unit and a communications area with a network interface. Furthermore, at least one functional object is provided, which is implemented by software or by software and hardware, and is capable of performing a function. This functional object is subdivided into a procedural element, and a communications element and possibly an operating element. In addition, at least one object manager is provided, which manages at least two functional objects, and in particular synchronizes their information exchange. This control structure is implemented by at least one data processing system, which processes the data from the functional objects and the object manager and which itself is designed as a task-oriented unit.

Replace the paragraph beginning at page 1, line 16, with the following paragraph:

No systematic process for starting a control implemented as a task-oriented unit is known from EP 0 524 344 B1. Note that EP 0 524 344 B1 corresponds to U.S. Patent No. 5,844,804, the entire contents of which are incorporated herein by reference.

Replace the paragraph beginning at page 1, line 18, with the following paragraph:

An object-oriented control for machine tools is known from EP 0 657 043 B1, wherein a number of objects is formed from classes of objects. Starting from the structure of a control disclosed in EP 0 524 344 B1, the definitely required classes of objects and the objects which are needed for implementing a conventional functionality of a control, are disclosed in EP 0 657 043 B1. Among these are, for example, classes of objects for types of processing, geometry,

kinematics and technological data, as well as types of control data and an object class of sequencing control. Any arbitrary number of objects can be formed from each class of objects, each of which contains its own data range, a messenger mechanism for communications with other objects and a procedural part for executing methods for processing, kinematics, geometry or technology. The user input is interpreted by the sequence control and leads to the activation of the selected objects. The selected objects communicate with each other and by this network-like linkage form a functional unit of the control which is capable of running.

Replace the paragraph beginning at page 1, line 31 with the following paragraph:

No systematic process for starting a numerical control is known from EP 0 657 043 B1, either. Note that EP 0 657 043 B1 corresponds to U.S. Patent No. 5,506,787, the entire contents of which are incorporated herein by reference.

Replace the paragraph portion beginning at page 2, line 1 of the translation and as amended in lines 1-8 of replacement page 3, with the following paragraph:

A CNC control system is known from EP 0 717 866 B1, which contains an object-oriented program in which objects exchange object-oriented information. The objects are divided into classes in the object-oriented program, for example a process class, which contains objects for work processes such as drilling, thread-cutting, reaming, etc., which are performed by machine components. Here, a class always contains similar objects, i.e. objects which agree in their basic structure. Based on the uniform basic structure of the objects in a class there is the possibility that, in the course of creating new objects, selected properties of the respective class are passed on to the new object. For example, a drilling object has depth and a diameter, which can be inherited by

another object of the process class, for example a thread- cutting object, which also has depth and a diameter. Another object class has machine components, such as a spindle, shafts, a turntable, etc., for example. Moreover, object classes are provided for the kernel, with a motion and a logic controller as objects, for platform services, the operating system and the device drivers. During
5 operation of the control it is necessary for messages to be exchanged between the individual objects. For example, in a drilling process a message regarding the number of revolutions of the drill is transmitted from the drill object to the object called spindle, furthermore messages regarding the position of the hole are transmitted to the objects of participating shafts, etc. Here, a standard interface is provided for the messages, so that they have a universal structure and can be designed independently of the involved objects. This standard interface for message exchanges between objects is implemented in connection with an object receiving or transmitting messages regarding the movement by a software kernel, which is intended to operate in real time and receives and transmits messages.

Replace the paragraph beginning at replacement page 3, line 9 with the following paragraph:

A method for a synchronized start-up of a numerical control is not disclosed in EP 0 717 866 B1. Note that EP 0 717 866 B1 corresponds to U.S. Patent No. 5,453,933, the entire contents of which are incorporated herein by reference.

Replace the paragraph beginning at replacement page 3, line 14 with the following paragraph:

A method for a synchronized start-up of a numerical control with hierarchically arranged

objects is not disclosed in DE 195 20 747 A1. Instead, DE 195 20 747 A1 describes a system of several object manager components, wherein object managers, which are to be initialized in several phases, are locally installed. Initialization takes place in several steps, wherein a new step is only permitted after the previous step has been executed.

5

Replace the paragraph beginning at replacement page 3, line 16 with the following heading and paragraph:

SUMMARY AND OBJECTS OF THE INVENTION

It is therefore an object of the present invention to cite a method for a start of an application of a numerical control wherein their processes, threads and modules are embedded into a structure and are initialized in a synchronized manner, and wherein thereafter a transition to a normal operation is made. It is intended to assure that in the course of transitioning to normal operations the communications channels, by modules, which are parts of different threads or processes, communicate with each other, are provided in particular.

Replace the paragraph beginning at replacement page 3, line 23, with the following paragraph:

This object is attained by a method for the synchronized start-up of an application of a numerical control of a mechanized element that includes sequentially performing several initialization steps for several objects of an application, wherein several first objects are hierarchically subordinated to the application, and at least one second object is hierarchically subordinated to each of the several first objects, initializing the several first objects and the at least one second object, initializing the several first objects via the application and initializing the at

least one second object via the first object. Transmitting a first execution report to one of the several first objects and transmitting a second execution report to the application via the first objects.

5 Between lines 23 and 24 of replacement page 3, insert the following new paragraph:

In accordance with the present invention, execution messages regarding the performed initialization steps of the individual objects are passed on to the respectively hierarchically higher objects. This so-called looping of the execution messages has the advantage that a subordinated object only needs to communicate with the object that has created it. In other words, the created object does not need to know its larger surroundings or environment. Because of this, it is particularly simple to change the object structure, in particular to complement it. During the start-up process, the object is almost immediately aware in which branch an error has occurred. It is therefore not absolutely necessary that the application know the entire structure in its hierarchical design.

Delete the paragraph beginning at line 24 on replacement page 3.

Replace the paragraph portion beginning at page 3, line 26-28 of replacement page 3 and continuing from lines 1 to 11 of page 4 of the translation, with the following paragraph:

20 The method in accordance with the present invention has the advantage that the software of a numerical control for a machine tool in the form of an application with the associated processes, threads and modules is started in a synchronized manner up to the operational readiness of the control. Here, in a first step the application is loaded into the main memory in the form of one or

several executable files, and the individual objects, for example processes, threads and modules, are created. As soon as all objects have been created, the objects are initialized, for example with parameter values, in a second step. As soon as this initializing step is completed, further initializing steps can follow until, in a last initializing step, the entire application is released for operation. Thus, all applications, processes, threads and modules have been created, initialized and the control is ready for operation in an error-free start-up. If errors should occur in an object, following such a synchronized start-up it is known which processes, threads or modules are wrong. A further advantage lies in that, if this does not hold true in connection with a required application, a process, a thread or a module, the user is informed.

Replace the paragraph beginning at page 4, line 12, with the following paragraph:

The present invention will be explained in greater detail in what follows by drawings.

Shown are in:

Replace the paragraph beginning on page 4, line 14 with the following heading and paragraph:

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 schematically shows an embodiment of circuit in accordance with the present invention; and

Replace the paragraph beginning at page 4, line 16 with the following paragraph:

FIG. 2 schematically shows an embodiment of a structure of an application implemented by the circuit of FIG. 1 according to the present invention.

Replace the paragraph beginning at page 4, line 17 with the following paragraph:

The paragraph invention is intended to be explained in what follows by a simple numerical control for a machine tool.

5

Replace the paragraph beginning on page 4, line 19 with the following heading and paragraph:

DESCRIPTION OF THE PREFERRED EMBODIMENT(S) OF THE INVENTION

In accordance with FIG. 1, the control arrangement has at least one bulk storage 1, which is connected with a processor 3 via a bidirectional bus 2. In this way the processor 3 can have a read and write access to the bulk storage 1. The bulk storage 1 can be implemented by a magnetic or optical memory or a combination of several different memories. The bulk storage 1 can have exchangeable storage media, for example compact disks. Via the bus 2, a bidirectional connection also exists with a main memory 4, which is designed as a write/read memory and has a shorter access time than the bulk storage 1. The main memory 4 can also be designed - at least partially - integrated into the processor 3. A bidirectional connection furthermore exists via the bus 2 with a fixed value memory 5 which, in contrast to the main memory 4, only has a small memory capacity and is designed as a read/write memory. A regulating assembly 6 is furthermore connected with the bus, by which the drive motors for the shafts and the spindle are connected in turn.

20

Replace the paragraph beginning at page 4, line 32 with the following paragraph:

So that the processor 3 can control the various functions of the machine tool, it is necessary to make the software of a numerical control for the machine tool available in the main memory 4,

which is identified as application 10 in what follows and is processed by the processor 3. Since as a rule the main memory 4 is not a permanent memory, an executable file is stored in the bulk storage 1, which permits the creation of the entire application 10 in the main memory 4 by the processor 3 as soon as the control arrangement has been connected to the supply voltage. The creation of the application 10 from the file stored in the bulk storage 1 is called initialization in what follows and usually takes place in several steps.

Replace the paragraph beginning at page 5, line 26 with the following paragraph:

As already explained, the software of the numerical control of the machine tool is called application 10, it has the function of the machine control or the function of a programming location. The processes 11 to 13, which are subordinated to the application 10, for example, have the function of interpolation, of the stored program programmable control SPS, or of the advance geometry calculation. The threads 14 to 17 subordinated to the processes 11 to 13 have the function of making central processing unit (CPU) time available to the processes 11 to 13, i.e. to distribute the CPU time. CPU time and data are required for an application 10, the data are made available in the form of memory space by the processes 11 to 13, and the CPU time by the associated thread 14 to 17. The modules 18 to 21 subordinated to the threads 14 to 17 divide a process 11 into logical sections, for example a coordinate transformation, from a logical viewpoint.

Replace the paragraph beginning at page 6, line 5 with the following paragraph:

In order to implement such a structure of the application 10 in a particularly simple form in accordance with program technology, an object-oriented programming language is advantageously

selected. By the object-oriented programming language the application 10, each process 11 to 13, each thread 14 to 17 and each module 18 to 21 are designed as an object. The objects 10 to 21 can be synchronized, can create further objects 11 to 21, and can manage the created objects 11 to 21. Such a structure, and the mentioned functions, can basically also be implemented by [means of] a non-object-oriented programming language.

Replace the paragraph beginning at page 6, line 13 with the following paragraph:

When the control arrangement is connected with the current supply, first its operating system software is loaded into the main memory 4 of the control arrangement, and subsequently the start program of the application 10. Thereafter this start program is executed and controls the synchronized creation of at least the further processes 11 to 13. It is also alternatively possible to control the creation of the entire structure 10 to 21 by the start program.

Replace the paragraph beginning at page 7, line 11 with the following paragraph:

This results in a tree-like structure of the objects 10 to 21, at whose tip the application 10 is located on the highest hierarchical level. The application 10 is subdivided into processes 11 to 13, which in turn includes threads 14 to 17. The modules 18 to 21, which are assigned to the threads 14 to 17, are located on the lowest hierarchical level.

Replace the paragraph beginning at page 7, line 16 with the following paragraph:

The linkage between the modules 18 to 21, the threads 14 to 17 and the processes 11 to 13, up to the application 10 is implemented by communications channels 30. Since the modules each have a defined functionality, defined communications channels 31 are required for each module 18

to 21, through which the data are transmitted from or to another module 18 to 21.

Replace the paragraph beginning at page 7, line 21 with the following paragraph:

5 In an embodiment in accordance with the present invention, each object 10 to 21 has, inter alia, a program element which is used for initializing this object 10 to 21 and which executes individual steps for initializing each object 10 to 21. These individual steps include the object 10 to 21 initiates the initialization of the objects 11 to 21, which are hierarchically subordinate to it, waits for the execution report of all subordinate objects 11 to 21 and then, following its own correct initialization, forwards an execution report to the higher-ranking object 10 to 21. Further initializing steps can additionally take place by the program element.

Replace the paragraph beginning at page 7, line 29 with the following paragraph:

This assignment of program elements can be implemented in a particularly advantageous manner by an object-oriented programming language.

Replace the paragraph beginning at page 9, line 13 with the following paragraph:

20 The possibility of performing further initializing steps exists, which basically run identically to the ones described so far. The execution of a program element for creating a defined function is always triggered by the hierarchically higher-ranking objects 10 to 17, and thereafter a waiting period takes place until an execution report is received from all subordinated objects 11 to 21, before an execution report is transmitted to a higher-ranking object 10 to 17. The next step of the initialization takes place only after the application 10 has received the execution reports from all assigned processes 11 to 13. By this it is assured that the initialization takes place

synchronously and that all objects 10 to 21 assume a defined state at a defined time.

Replace the paragraph beginning at page 9, line 25 with the following paragraph:

Starting of the application 10 takes place as the last initialization step. In the course of
5 this, each object 10 to 21 is given a signal that the initialization has been completed and the
regular operation of the control starts from here. From this an object 10 to 21 recognizes that it is
to perform the regular processing of data received through the communications channels. No
execution report to the higher-ranking object 10 to 17 is absolutely required in this last step, since
the correct initialization has already been signaled by the previous execution reports, and the
synchronicity required for initialization is no longer needed.

Replace the paragraph beginning at page 11, line 28 with the following paragraph:

Now the initialization of the communications channels 30, 31 between the objects 10 to 21
follows and proceeds in accordance with the same system as the creation of the structure 10 to 21
of the software. In the process, at least logical communication channels 30, 31 are defined for the
application object 10 and each process object, thread object and module object 11 to 21, by which
they can communicate with already created objects. In this case a communications channel 30, 31
can also be considered to be an object. For the creation of the communications channels 30, 31,
information regarding the further objects with which messages are to be exchanged is already
20 stored in the application object, the process object, thread object and module object 11 to 21.
Communications channels 30, 31 to these objects are then created.

Replace the paragraph beginning at page 13, line 10 with the following paragraph:

Advantageously the individual steps sequentially includes the creation of the process 11 to 13, of the threads 14 to 17 and of the modules 18 to 21, until all required objects 11 to 21 have been created, thereafter the creation of the communications channels 30, 31 by the respective receiver, the opening of the communications channels by the transmitter, the parameterization of the modules 18 to 21 and the transition to normal operations of the numerical control.

After line 15 at page 13 insert the following paragraph:

The invention may be embodied in other forms than those specifically disclosed herein without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive, and the scope of the invention is commensurate with the appended claims rather than the foregoing description.

Replace the paragraph beginning at page 14, line 1, with the following paragraph:

We Claim:

After page 16 add a new page 17 to read as follows:

Abstract of the Disclosure

A method for the synchronized start-up of an application of a numerical control of a mechanized element that includes sequentially performing several initialization steps for several objects of an application, wherein several first objects are hierarchically subordinated to the application, and at least one second object is hierarchically subordinated to each of the several first objects, initializing the several first objects and the at least one second object, initializing the several first objects via the application and initializing the at least one second object via the first

object. Transmitting a first execution report to one of the several first objects and transmitting a second execution report to the application via the first objects.

In the Claims:

5 Please cancel claims 1-10, as amended, without prejudice and add claims 11-42 as follows:

11. (New) A method for the synchronized start-up of an application of a numerical control of a mechanized element, the method comprising:

sequentially performing several initialization steps for several objects of an application, wherein several first objects are hierarchically subordinated to said application, and at least one second object is hierarchically subordinated to each of said several first objects;

performing one of said initialization steps in said several first objects and said at least one second object;

performing one of said initialization steps in said several first objects via said application;

performing one of said initialization steps in said at least one second object via at least one of said several first objects;

transmitting a first execution report to one of said several first objects; and

transmitting a second execution report to said application via said several first objects.

12. (New) The method of claim 11, wherein said transmitting said first execution report occurs following correct execution of said performing one of said initialization steps in said at least one second object.

13. (New) The method of claim 11, wherein said transmitting said second execution report occurs following correct execution of said performing one of said initialization steps in said several first objects.

5

14. (New) The method of claim 12, wherein said transmitting said second execution report occurs following correct execution of said performing one of said initialization steps in said several first objects.

15. (New) The method of claim 11, wherein said mechanized element comprises a machine tool.

16. (New) The method of claim 11, wherein said mechanized element comprises a robot.

17. (New) The method of claim 11, further comprising executing a starting program.

18. (New) The method of claim 17, wherein said executing said starting program comprises loading at least a part of data of said application into a memory.

19. (New) The method of claim 17, wherein said starting program comprises information regarding said application and a chronological sequence of said several initialization steps.

20. (New) The method of claim 11, wherein a step involved in said performing one of said initialization steps in said several first objects is stored in one of said several first objects.

5 21. (New) The method of claim 11, wherein a step involved in said performing one of said initialization steps in said at least one second object is stored in said at least one second object.

22. (New) The method of claim 20, wherein a step involved in said performing one of said initialization steps in said at least one second object is stored in said at least one second object.

23. (New) The method of claim 11, further comprising initializing a third object that is hierarchically subordinated to said at least one second object, wherein said at least one second object initiates said initializing said third object.

24. (New) The method of claim 23, further comprising:
transmitting a third execution message to said at least one second object via said third object; and

20 transmitting a fourth execution message to said several first objects via said at least one second object.

25. (New) The method of claim 11, wherein said several first objects are processes of said application.

5 26. (New) The method of claim 11, wherein said at least one second object is a thread that makes central processor unit time available.

27. (New) The method of claim 25, wherein said at least one second object is a thread that makes central processor unit time available.

28. (New) The method of claim 24, wherein said several first objects are processes of said application and said at least one second object is a thread that makes central processor unit time available.

29. (New) The method of claim 11, further comprising implementing a definite function of a numeric control in each of said several first objects.

30. (New) The method of claim 28, further comprising implementing a definite function of a numeric control in each of said several first objects.

31. (New) The method of claim 11, further comprising creating individual, essentially completed functions of said application in the form of said several first objects.

32. (New) The method of claim 11, further comprising creating individual, essentially completed functions of said application in the form of said at least one second object.

5 33. (New) The method of claim 11, further comprising creating communications channels between said several first objects and said at least one second object.

34. (New) The method of claim 11, further comprising assigning a parameter to a variable in said several first objects.

35. (New) The method of claim 11, further comprising assigning a parameter to a variable in said at least one second object.

36. (New) The method of claim 11, further comprising releasing a regular operation of said application.

37. (New) A method for the synchronized start-up of an application of a numerical control of a mechanized element, the method comprising:

sequentially performing several initialization steps for several objects of an
20 application, wherein several first objects are hierarchically subordinated to said application, and at least one second object is hierarchically subordinated to each of said several first objects;
performing one of said initialization steps in said several first objects via said

application;

performing said one of said initialization steps in said at least one second object via
at least one of said several first objects;

transmitting a first execution report via said initialized at least one second object to
5 one of said several first objects;

transmitting a second execution report to said application via said initialized several
first objects that have received said first execution report; and

further initializing said several first objects and said at least one second object via
said application after said application receives said second execution report and after said
performing said one of said initialization steps in said several first objects and said at least one
second object.

38. (New) The method of claim 37, wherein said mechanized element comprises a
machine tool.

39. (New) The method of claim 37, wherein said mechanized element comprises a
robot.

40. (New) The method of claim 37, further comprising performing, via said at least
20 one second object, said one of said initialization steps in a third object that is hierarchically
subordinated to said at least one second object;

transmitting a third execution message to said at least one second object via said

third object; and

transmitting a fourth execution message to said several first objects via said second object after said third object receives said third execution message and after said initializing said third object.

5

41. (New) The method of claim 40, wherein said several first objects are processes of said application and said at least one second object is a thread that makes central processor unit time available.

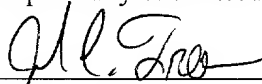
42. (New) The method of claim 37, further comprising creating communications channels between said several first objects and said at least one second object.

Please note that new claims 11-42 are being presented to provide additional coverage regarding a method for the synchronized start-up of an application of a numerical control of a mechanical object. In addition, since the original claims reflect a literal translation of the claims of the corresponding Patent Cooperation Treaty, there is a need to stylize their language to everyday English and to use U.S. patent terminology. Accordingly, the cancellation of original claims 1-10 and the addition of new claims 11-42 are not being presented for reasons of

patentability as defined in Festo Corporation v. Shoketsu Kinzoku Kogyo Kabushiki Co., Ltd.,

234 F.3d 558, 56 USPQ2d 1865 (Fed. Cir. 2000).

Respectfully submitted,



John C. Freeman

Registration No. 34,483

Attorney for Applicants

BRINKS HOFER
GILSON & LIONE
P.O. Box 10395
Chicago, Illinois 60610
(312) 321-4200

Dated: November 28, 2001

Marked up Version of Specification

5 The present invention relates to a method for the synchronized start-up of a numerical control [in accordance with the preamble of claim 1].

EP 0 524 344 B1 discloses a configurable machine tool control [consisting of] having several task-oriented units, for example a numeric and a memory-programmable control element, an operating unit and a communications area with a network interface. Furthermore, at least one
10 functional object is provided, which is implemented by [means of] software or by [means of] software and hardware, and is capable of performing a function. This functional object is subdivided into a procedural element, and a communications element and possibly an operating element. In addition, at least one object manager is provided, which manages at least two functional objects, and in particular synchronizes their information exchange. This control
15 structure is implemented by [means of] at least one data processing system, which processes the data from the functional objects and the object manager and which itself is designed as a task-oriented unit.

No systematic process for starting a control implemented as a task-oriented unit is known
20 from EP 0 524 344 B1. Note that EP 0 524 344 B1 corresponds to U.S. Patent No. 5,844,804, the entire contents of which are incorporated herein by reference.

An object-oriented control for machine tools is known from EP 0 657 043 B1, wherein a

number of objects is formed from classes of objects. Starting from the structure of a control disclosed in EP 0 524 344 B1, the definitely required classes of objects and the objects which are needed for implementing a conventional functionality of a control, are disclosed in EP 0 657 043 B1. Among these are, for example, classes of objects for types of processing, geometry, kinematics and technological data, as well as types of control data and an object class of sequencing control. Any arbitrary number of objects can be formed from each class of objects, each of which contains its own data range, a messenger mechanism for communications with other objects and a procedural part for executing methods for processing, kinematics, geometry or technology. The user input is interpreted by the sequence control and leads to the activation of the selected objects. The selected objects communicate with each other and by [means of] this network-like linkage form a functional unit of the control which is capable of running.

No systematic process for starting a numerical control is known from EP 0 657 043 B1, either. Note that EP 0 657 043 B1 corresponds to U.S. Patent No. 5,506,787, the entire contents of which are incorporated herein by reference.

A CNC control system is known from EP 0 717 866 B1, which contains an object-oriented program in which objects exchange object-oriented information. The objects are divided into classes in the object-oriented program, for example a process class, which contains objects for work processes such as drilling, thread-cutting, reaming, etc., which are performed by machine components. Here, a class always contains similar objects, i.e. objects which agree in their basic structure. Based on the uniform basic structure of the objects in a class there is the possibility that,

in the course of creating new objects, selected properties of the respective class are passed on to the new object. For example, a drilling object has depth and a diameter, which can be inherited by another object of the process class, for example a thread-cutting object, which also has depth and a diameter. Another object class has machine components, such as a spindle, shafts, a turntable, etc., for example. Moreover, object classes are provided for the kernel, with a motion and a logic controller as objects, for platform services, the operating system and the device drivers. During operation of the control it is necessary for messages to be exchanged between the individual objects. For example, a drilling object has depth and a diameter, which can be inherited by another object of the process class, for example a thread-cutting object, which also has depth and a diameter. Another object class has machine components, such as a spindle, shafts, a turntable, etc., for example. Moreover, object classes are provided for the kernel, with a motion and a logic controller as objects, for platform services, the operating system and the device drivers. During operation of the control it is necessary for messages to be exchanged between the individual objects. For example, in a drilling process a message regarding the number of revolutions of the drill is transmitted from the drill object to the object called spindle, furthermore messages regarding the position of the hole are transmitted to the objects of participating shafts, etc. Here, a standard interface is provided for the messages, so that they have a universal structure and can be designed independently of the involved objects. This standard interface for message exchanges between objects is implemented in connection with an object receiving or transmitting messages regarding the movement by [means of] a software kernel, which is intended to operate in real time and receives and transmits messages.

A method for a synchronized start-up of a numerical control is not disclosed [there] in EP 0 717 866 B1. Note that EP 0 717 866 B1 corresponds to U.S. Patent No. 5,453,933, the entire contents of which are incorporated herein by reference.

5 A method for a synchronized start-up of a numerical control with hierarchically arranged objects is not disclosed [there] in DE 195 20 747 A1. Instead, DE 195 20 747 A1 describes a system of several object manager components, wherein object managers, which are to be initialized in several phases, are locally installed. Initialization takes place in several steps, wherein a new step is only permitted after the previous step has been executed.

SUMMARY AND OBJECTS OF THE INVENTION

It is therefore [the] an object of the present invention to cite a method for a start of an application of a numerical control wherein their processes, threads and modules are embedded into a structure and are initialized in a synchronized manner, and wherein thereafter a transition to a normal operation is made. It is intended to assure that in the course of transitioning to normal operations the communications channels, by [means of which] modules, which are parts of different threads or processes, communicate with each other, are provided in particular.

This object is attained by [means of a method having the features recited in claim 1] a
20 method for the synchronized start-up of an application of a numerical control of a mechanized element that includes sequentially performing several initialization steps for several objects of an application, wherein several first objects are hierarchically subordinated to the application, and at

5
least one second object is hierarchically subordinated to each of the several first objects,
initializing the several first objects and the at least one second object, initializing the several first
objects via the application and initializing the at least one second object via the first object.
Transmitting a first execution report to one of the several first objects and transmitting a second
execution report to the application via the first objects.

20
The method in accordance with the present invention has the advantage that the software of
a numerical control for a machine tool in the form of an application with the associated processes,
threads and modules is started in a synchronized manner up to the operational readiness of the
control. Here, in a first step the application is loaded into the main memory in the form of one or
several executable files, and the individual objects, for example processes, threads and modules,
are created. As soon as all objects have been created, the objects are initialized, for example with
parameter values, in a second step. As soon as this initializing step is completed, further
initializing steps can follow until, in a last initializing step, the entire application is released for
operation. Thus, all applications, processes, threads and modules have been created, initialized
and the control is ready for operation in an error-free start-up. If errors should occur in an object,
following such a synchronized start-up it is known which processes, threads or modules are
wrong. A further advantage lies in that, if this does not hold true in connection with a required
application, a process, a thread or a module, the user is informed.

20
The present invention will be explained in greater detail in what follows by [means of]
drawings. Shown are in:

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1[, a possible form of an implementation in accordance with circuit technology of a control of] schematically shows an embodiment of circuit in accordance with the present invention; and[,]

FIG. 2[, a possible] schematically shows an embodiment of a structure of an application implemented by the circuit of FIG. 1 according to the present invention.

The paragraph invention is intended to be explained in what follows by [means of] a simple numerical control for a machine tool.

DESCRIPTION OF THE PREFERRED EMBODIMENT(S) OF THE INVENTION

In accordance with FIG. 1, the control arrangement has at least one bulk storage 1, which is connected with a processor 3 via a bidirectional bus 2. In this way the processor 3 can have a read and write access to the bulk storage 1. The bulk storage 1 can be implemented by [means of] a magnetic or optical memory or a combination of several different memories. The bulk storage 1 can have exchangeable storage media, for example compact disks. Via the bus 2, a bidirectional connection also exists with a main memory 4, which is designed as a write/read memory and has a shorter access time than the bulk storage 1. The main memory 4 can also be designed - at least partially - integrated into the processor 3. A bidirectional connection furthermore exists via the bus 2 with a fixed value memory 5 which, in contrast to the main memory 4, only has a small

memory capacity and is designed as a read/write memory. A regulating assembly 6 is furthermore connected with the bus, by [means of] which the drive motors for the shafts and the spindle are connected in turn.

5 So that the processor 3 can control the various functions of the machine tool, it is necessary to make the software of a numerical control for the machine tool available in the main memory 4, which is identified as application 10 in what follows and is processed by the processor 3. Since as a rule the main memory 4 is not a permanent memory, an executable file is stored in the bulk storage 1, which permits the creation of the entire application 10 in the main memory 4 by [means of] the processor 3 as soon as the control arrangement has been connected to the supply voltage. The creation of the application 10 from the file stored in the bulk storage 1 is called initialization in what follows and usually takes place in several steps.

As already explained, the software of the numerical control of the machine tool is called application 10, it has the function of the machine control or the function of a programming location. The processes 11 to 13, which are subordinated to the application 10, for example, have the function of interpolation, of the stored program programmable control SPS, or of the advance geometry calculation. The threads 14 to 17 subordinated to the processes 11 to 13 have the function of making central processing unit (CPU) time available to the processes 11 to 13, i.e. to distribute the CPU time. CPU time and data are required for an application 10, the data are made available in the form of memory space by the processes 11 to 13, and the CPU time by the associated thread 14 to 17. The modules 18 to 21 subordinated to the threads 14 to 17 divide a

process 11 into logical sections, for example a coordinate transformation, from a logical viewpoint.

In order to implement such a structure of the application 10 in a particularly simple form in accordance with program technology, an object-oriented programming language is advantageously selected. By [means of] the object-oriented programming language the application 10, each process 11 to 13, each thread 14 to 17 and each module 18 to 21 are designed as an object. The objects 10 to 21 can be synchronized, can create further objects 11 to 21, and can manage the created objects 11 to 21. Such a structure, and the mentioned functions, can basically also be implemented by [means of] a non-object-oriented programming language.

When the control arrangement is connected with the current supply, first its operating system software is loaded into the main memory 4 of the control arrangement, and subsequently the start program of the application 10. Thereafter this start program is executed and controls the synchronized creation of at least the further processes 11 to 13. It is also alternatively possible to control the creation of the entire structure 10 to 21 by [means of] the start program.

This results in a tree-like structure of the objects 10 to 21, at whose tip the application 10 is located on the highest hierarchical level. The application 10 is subdivided into processes 11 to 13, which in turn [consist of] includes threads 14 to 17. The modules 18 to 21, which are assigned to the threads 14 to 17, are located on the lowest hierarchical level.

The linkage between the modules 18 to 21, the threads 14 to 17 and the processes 11 to 13, up to the application 10 is implemented by [means of] communications channels 30. Since the modules each have a defined functionality, defined communications channels 31 are required for each module 18 to 21, through which the data are transmitted from or to another module 18 to 21.

5

In an embodiment in accordance with the present invention, each object 10 to 21 has, inter alia, a program element which is used for initializing this object 10 to 21 and which executes individual steps for initializing each object 10 to 21. These individual steps at least [consist in that] include the object 10 to 21 initiates the initialization of the objects 11 to 21, which are hierarchically subordinate to it, waits for the execution report of all subordinate objects 11 to 21 and then, following its own correct initialization, forwards an execution report to the higher-ranking object 10 to 21. Further initializing steps can additionally take place by [means of] the program element.

This assignment of program elements can be implemented in a particularly advantageous manner by [means of] an object-oriented programming language.

The possibility of performing further initializing steps exists, which basically run identically to the ones described so far. The execution of a program element for creating a defined function is always triggered by the hierarchically higher-ranking objects 10 to 17, and thereafter a waiting period takes place until an execution report is received from all subordinated objects 11 to 21, before an execution report is transmitted to a higher-ranking object 10 to 17. The next step of

the initialization takes place only after the application 10 has received the execution reports from all assigned processes 11 to 13. By [means of] this it is assured that the initialization takes place synchronously and that all objects 10 to 21 assume a defined state at a defined time.

5 Starting of the application 10 takes place as the last initialization step. In the course of this, each object 10 to 21 is given a signal that the initialization has been completed and the regular operation of the control starts from here. From this an object 10 to 21 recognizes that it is to perform the regular processing of data received through the communications channels. No execution report to the higher-ranking object 10 to 17 is absolutely required in this last step, since the correct initialization has already been signaled by [means of] the previous execution reports, and the synchronicity required for initialization is no longer needed.

Now the initialization of the communications channels 30, 31 between the objects 10 to 21 follows and proceeds in accordance with the same system as the creation of the structure 10 to 21 of the software. In the process, at least logical communication channels 30, 31 are defined for the application object 10 and each process object, thread object and module object 11 to 21, by [means of] which they can communicate with already created objects. In this case a communications channel 30, 31 can also be considered to be an object. For the creation of the communications channels 30, 31, information regarding the further objects with which messages are to be
20 exchanged is already stored in the application object, the process object, thread object and module object 11 to 21. Communications channels 30, 31 to these objects are then created.

Advantageously the individual steps sequentially [comprise] includes the creation of the process 11 to 13, of the threads 14 to 17 and of the modules 18 to 21, until all required objects 11 to 21 have been created, thereafter the creation of the communications channels 30, 31 by the respective receiver, the opening of the communications channels by the transmitter, the
5 parameterization of the modules 18 to 21 and the transition to normal operations of the numerical control.

We Claim:

Method for the Synchronized Start-up of a Numerical Control

The invention relates to a method for the synchronized start-up of a numerical control in accordance with the preamble of claim 1.

5 EP 0 524 344 B1 discloses a configurable machine tool control consisting of several task-oriented units, for example a numeric and a memory-programmable control element, an operating unit and a communications area with a network interface. Furthermore, at least one functional object is provided, which is implemented by means of software or by means of software and hardware, and is capable of performing a function.

10 This functional object is subdivided into a procedural element, and a communications element and possibly an operating element. In addition, at least one object manager is provided, which manages at least two functional objects, and in particular synchronizes their information exchange. This control structure is implemented by means of at least one data processing system, which processes the data from the functional objects and the
15 object manager and which itself is designed as a task-oriented unit.

No systematic process for starting a control implemented as a task-oriented unit is known from EP 0 524 344 B1.

An object-oriented control for machine tools is known from EP 0 657 043 B1, wherein a number of objects is formed from classes of objects. Starting from the structure
20 of a control disclosed in EP 0 524 344 B1, the definitely required classes of objects and the objects which are needed for implementing a conventional functionality of a control, are disclosed in EP 0 657 043 B1. Among these are, for example, classes of objects for types of processing, geometry, kinematics and technological data, as well as types of control data and an object class of sequencing control. Any arbitrary number of objects can be formed
25 from each class of objects, each of which contains its own data range, a messenger mechanism for communications with other objects and a procedural part for executing methods for processing, kinematics, geometry or technology. The user input is interpreted by the sequence control and leads to the activation of the selected objects. The selected objects communicate with each other and by means of this network-like linkage form a
30 functional unit of the control which is capable of running.

No systematic process for starting a numerical control is known from EP 0 657 043 B1, either.

A CNC control system is known from EP 0 717 866 B1, which contains an object-oriented program in which objects exchange object-oriented information. The objects are divided into classes in the object-oriented program, for example a process class, which contains objects for work processes such as drilling, thread- cutting, reaming, etc., which are performed by machine components. Here, a class always contains similar objects, i.e. objects which agree in their basic structure. Based on the uniform basic structure of the objects in a class there is the possibility that, in the course of creating new objects, selected properties of the respective class are passed on to the new object. For example, a drilling object has depth and a diameter, which can be inherited by another object of the process class, for example a thread- cutting object, which also has depth and a diameter. Another object class has machine components, such as a spindle, shafts, a turntable, etc., for example. Moreover, object classes are provided for the kernel, with a motion and a logic controller as objects, for platform services, the operating system and the device drivers. During operation of the control it is necessary for messages to be exchanged between the individual objects. For

example, in a drilling process a message regarding the number of revolutions of the drill is transmitted from the drill object to the object called spindle, furthermore messages regarding the position of the hole are transmitted to the objects of participating shafts, etc. Here, a standard interface is provided for the messages, so that they have a universal structure and can be designed independently of the involved objects. This standard interface for message exchanges between objects is implemented in connection with an object receiving or transmitting messages regarding the movement by means of a software kernel, which is intended to operate in real time and receives and transmits messages.

A method for a synchronized start-up of a numerical control is not disclosed there.

It is therefore the object of the present invention to cite a method for a start of an application of a numerical control wherein their processes, threads and modules are embedded into a structure and are initialized in a synchronized manner, and wherein thereafter a transition to a normal operation is made. It is intended to assure that in the course of transitioning to normal operations the communications channels, by means of which modules, which are parts of different threads or processes, communicate with each other, are provided in particular.

This object is attained by means of a method having the features recited in claim 1.

Further developments and advantageous embodiments of the method in accordance with the invention can be taken from the dependent claims.

The method in accordance with the invention has the advantage that the software of a numerical control for a machine tool in the form of an application with the associated processes, threads and modules is started in a synchronized manner up to the

operational readiness of the control. Here, in a first step the application is loaded into the main memory in the form of one or several executable files, and the individual objects, for example processes, threads and modules, are created. As soon as all objects have been created, the objects are initialized, for example with parameter values, in a second step.

5 As soon as this initializing step is completed, further initializing steps can follow until, in a last initializing step, the entire application is released for operation. Thus, all applications, processes, threads and modules have been created, initialized and the control is ready for operation in an error-free start-up. If errors should occur in an object, following such a synchronized start-up it is known which processes, threads or modules are wrong. A
10 further advantage lies in that, if this does not hold true in connection with a required application, a process, a thread or a module, the user is informed.

The invention will be explained in greater detail in what follows by means of drawings. Shown are in:

Fig. 1, a possible form of an implementation in accordance with circuit technology
15 of a control of the invention,

Fig. 2, a possible structure of an application.

The invention is intended to be explained in what follows by means of a simple numerical control for a machine tool.

In accordance with Fig. 1, the control arrangement has at least one bulk storage 1,
20 which is connected with a processor 3 via a bidirectional bus 2. In this way the processor 3 can have a read and write access to the bulk storage 1. The bulk storage 1 can be implemented by means of a magnetic or optical memory or a combination of several different memories. The bulk storage 1 can have exchangeable storage media, for example compact disks. Via the bus 2, a bidirectional connection also exists with a main memory
25 4, which is designed as a write/read memory and has a shorter access time than the bulk storage 1. The main memory 4 can also be designed - at least partially - integrated into the processor 3. A bidirectional connection furthermore exists via the bus 2 with a fixed value memory 5 which, in contrast to the main memory 4, only has a small memory capacity and is designed as a read/write memory. A regulating assembly 6 is furthermore connected
30 with the bus, by means of which the drive motors for the shafts and the spindle are connected in turn.

So that the processor 3 can control the various functions of the machine tool, it is

necessary to make the software of a numerical control for the machine tool available in the main memory 4, which is identified as application 10 in what follows and is processed by the processor 3. Since as a rule the main memory 4 is not a permanent memory, an executable file is stored in the bulk storage 1, which permits the creation of the entire application 10 in the main memory 4 by means of the processor 3 as soon as the control arrangement has been connected to the supply voltage. The creation of the application 10 from the file stored in the bulk storage 1 is called initialization in what follows and usually takes place in several steps.

First, a start program is executed, which contains information regarding the application 10 and the chronological sequence of the individual initialization steps, and which loads at least a part of the data of the application 10 from the bulk storage 1 into the main memory 4. The structure of the application 10 is created at the start of initialization. In the process, the data of the at least one file stored in the bulk storage 1 and being part of the application 10 are transferred into the main memory 4 and are processed. Objects 11 to 21, which are linked to each other, are created during processing.

In accordance with Fig. 2, an application 10 is subdivided into several processes 11 to 13, wherein each process 11 to 13 advantageously has its own address space in the main memory 4. However, several or all processes 11 to 13 can share a common address space. Each process 11 to 13 is subdivided into threads 14, 15, 16, 17 and has at least one thread 14, 15, 16, 17. The code processed in the threads 14 to 17 is divided into modules 18 to 21, a part of each of which is a defined data structure and in which at least one defined function of the control is implemented, for example to machine a pocket or to cut a thread, etc. The modules 18 to 21, threads 14 to 17 and processes 11 to 13, which are part of the application 10, are linked to each other in such a way that they can also exchange data past the limits of threads 14 to 17 and processes 11 to 13.

As already explained, the software of the numerical control of the machine tool is called application 10, it has the function of the machine control or the function of a programming location. The processes 11 to 13, which are subordinated to the application 10, for example, have the function of interpolation, of the stored program programmable control SPS, or of the advance geometry calculation. The threads 14 to 17 subordinated to the processes 11 to 13 have the function of making CPU time available to the processes 11 to 13, i.e. to distribute the CPU time. CPU time and data are required for an application

10, the data are made available in the form of memory space by the processes 11 to 13, and the CPU time by the associated thread 14 to 17. The modules 18 to 21 subordinated to the threads 14 to 17 divide a process 11 into logical sections, for example a coordinate transformation, from a logical viewpoint.

5 In order to implement such a structure of the application 10 in a particularly simple form in accordance with program technology, an object-oriented programming language is advantageously selected. By means of the object-oriented programming language the application 10, each process 11 to 13, each thread 14 to 17 and each module 18 to 21 are designed as an object. The objects 10 to 21 can be synchronized, can create further objects
10 11 to 21, and can manage the created objects 11 to 21. Such a structure, and the mentioned functions, can basically also be implemented by means of a non-object-oriented programming language.

When the control arrangement is connected with the current supply, first its operating system software is loaded into the main memory 4 of the control arrangement,
15 and subsequently the start program of the application 10. Thereafter this start program is executed and controls the synchronized creation of at least the further processes 11 to 13. It is also alternatively possible to control the creation of the entire structure 10 to 21 by means of the start program.

The data required by the start program are stored in a separate file, which must be
20 read in, or directly in the start program itself. If these data are stored in a separate file, the start program also loads this file into the main memory 4 and evaluates it. The data identify at least one executable file, which is stored in the bulk storage 1 and is to be loaded, for a process 11 to 13 of the application 10, and optionally transfer parameters of the processes 11 to 13, information for synchronizing the processes 11 to 13, and
25 information regarding the initializing steps to be performed during initialization. These executable files defined in this manner are loaded by the start program into the main memory 4 of the control and are evaluated. The processes 11 to 13 can be subsequently created.

Thus, in the course of processing the start program for the synchronized start-up of
30 the control, the files required for the application 10 are sequentially loaded into the main memory 4. Thereafter the processes 11 to 13, which constitute the application 10, are created, then the threads 14 to 17, and finally the data structures which are a part of the

modules 18 to 21. In this case further data are needed for creating the threads 14 to 17, which are a part of each process 11 to 13, and for the modules 18 to 21, which are a part of every thread 14 to 17. The data for creating the threads 14 to 17 can be stored in the start program, or in a file read in by the start program, in an executable file of the processes 11 to 13, or in a respectively read-in file.

This applies correspondingly also to the modules 18 to 21. Here, too, the data needed for their creation can be stored in the start program, or in a file read in by the start program, in an executable file, or in an individual file. Based on data regarding a module 18 to 21, the thread 14 to 17 as a part of the respective module 18 to 21 creates this module 18 to 21.

This results in a tree-like structure of the objects 10 to 21, at whose tip the application 10 is located on the highest hierarchical level. The application 10 is subdivided into processes 11 to 13, which in turn consist of threads 14 to 17. The modules 18 to 21, which are assigned to the threads 14 to 17, are located on the lowest hierarchical level.

The linkage between the modules 18 to 21, the threads 14 to 17 and the processes 11 to 13, up to the application 10 is implemented by means of communications channels 30. Since the modules each have a defined functionality, defined communications channels 31 are required for each module 18 to 21, through which the data are transmitted from or to another module 18 to 21.

In an embodiment in accordance with the invention, each object 10 to 21 has, inter alia, a program element which is used for initializing this object 10 to 21 and which executes individual steps for initializing each object 10 to 21. These individual steps at least consist in that the object 10 to 21 initiates the initialization of the objects 11 to 21, which are hierarchically subordinate to it, waits for the execution report of all subordinate objects 11 to 21 and then, following its own correct initialization, forwards an execution report to the higher-ranking object 10 to 21. Further initializing steps can additionally take place by means of the program element.

This assignment of program elements can be implemented in a particularly advantageous manner by means of an object-oriented programming language.

As soon as a module 18 to 21 has been correctly created and has been inserted into the object-oriented structure, the module 18 to 21 sends an execution report back to the

hierarchical thread 14 to 17, which is higher-ranking than it and which has created the module 18 to 21. From this the thread 14 to 17 determines that the module 18 to 21 has been created without errors. As soon as the thread 14 to 17 has received the execution report from each module 18 to 21 which it had created, and the thread 14 to 17 itself has also been created without errors, it also sends an execution report to the process 11 to 13 which is higher-ranking than it. As soon as each process 11 to 13 in turn has received an execution report of the correct creation from all threads 14 to 17 created by it, and the process 11 to 13 has also been correctly created, the process 11 to 13 sends an execution report to the higher-ranking application 10. In the represented example, only one module 18 to 21 each has been assigned to each thread 14 to 17, but several modules can also be assigned to each thread 14 to 17.

As soon as the application 10 has determined the correct creation of all processes 11 to 13 created by it on the basis of the received execution reports, the next step of the initialization takes place. All already created objects 11 to 21 must wait until then, so that no overlaps occur, for example because a first object tries to communicate with a second object, although the second object had not yet been created. The synchronization of the initialization is achieved by this waiting period, since all objects 11 to 21 are in a defined state at a defined time. Mechanisms, such as semaphores and barriers, or the like, are available for implementing this waiting period in a typical multi-tasking operating system.

In a second step of the initialization, the communications channels 30, 31 between the objects 10 to 21 are established. By causing the execution of the program element for the initialization of the communication channels 30 provided in the respective process 11 to 13, the communications channels 30 of one process 11 to 13 are established. This is also continued, corresponding to the manner already described in connection with the creation of the structure, for the threads 14 to 17 and modules 18 to 21, until all modules 18 to 21 have created their communications channels 30, 31.

Following the successful creation of all communications channels 30, 31 by a module 18 to 21, the latter returns an execution report to the thread 14 to 17 which initiated this. As soon as a thread 14 to 17 has received an execution report from all assigned modules 18 to 21, and also has itself successfully created all communications channels 30, the thread 14 to 17 also sends an execution report to the higher-ranking process 11 to 13. Once a process 11 to 13 has received an execution report from all

assigned threads 14 to 17, and also has itself correctly created all communications channels 30, it sends an execution report to the higher-ranking application 10. As soon as the application 10 has received an execution report from all assigned processes 11 to 13, this second step of the initialization is also completed.

5 Since it is necessary to exchange the most varied data as rapidly as possible between the objects 10 to 21 during the operation of the control, the communications channels can be differently implemented in accordance with the requirements. For example, a communications channel between two modules 18 to 21 of the same thread 14 to 17 can be implemented in the form of a memory common to both modules 18 to 21, through which data are exchanged; a communications channel 31 between modules 18 to 21 in different processes 11 to 13 must be able to transmit data from one address space to another.

10 The possibility of performing further initializing steps exists, which basically run identically to the ones described so far. The execution of a program element for creating a defined function is always triggered by the hierarchically higher-ranking objects 10 to 17, and thereafter a waiting period takes place until an execution report is received from all subordinated objects 11 to 21, before an execution report is transmitted to a higher-ranking object 10 to 17. The next step of the initialization takes place only after the application 10 has received the execution reports from all assigned processes 11 to 13. By means of this it is assured that the initialization takes place synchronously and that all objects 10 to 21 assume a defined state at a defined time.

20 For example, parameters can be transmitted to individual objects 10 to 21 in these further initializing steps. The basic operation takes place as has already been described for the previous initializing step.

25 Starting of the application 10 takes place as the last initialization step. In the course of this, each object 10 to 21 is given a signal that the initialization has been completed and the regular operation of the control starts from here. From this an object 10 to 21 recognizes that it is to perform the regular processing of data received through the communications channels. No execution report to the higher-ranking object 10 to 17 is absolutely required in this last step, since the correct initialization has already been signaled by means of the previous execution reports, and the synchronicity required for initialization is no longer needed.

A simple example of an initialization is represented in Fig. 2. Here, first the loading program for loading the start program is executed, so that the start program is in the main memory 4. If the data are not already contained in the start program, a file is subsequently loaded by the start program, which contains information regarding the technical program structure of the entire application 10, as well as regarding one or several files in which the entire application 10 has been stored in the bulk storage 10. This information is employed by the start program in such a way that all files containing information regarding the application 10 are loaded into the main memory 4 and the application 10 is created by the start program.

It is known to the application 10 created in this way that the processes 11, 12 and 13 are assigned to the application 10. Therefore the structure of the processes 11, 12 and 13 is created by the application 10. This is accomplished in that the application 10 loads at least one executable file into the main memory 4 and starts the execution of this at least one file. All processes 11, 12 and 13 are created in this way.

The newly created processes 11, 12 and 13 contain information regarding which threads 14 to 17 are assigned to the respective process 11, 12 and 13, so that the creation of the threads 14, 15, 16 and 17 is initiated by the processes 11 to 13.

In turn, the threads 14 to 17 contain information as to whether further modules 18 to 21 have been assigned to them. The modules 18 to 21 are created in this way. The modules 18 to 21 constitute the lowest hierarchical level in the control software of the machine tool, so that they do not contain any further information regarding assigned objects.

Which further objects 11 to 21 are assigned to each object 10 to 17 is stored in the latter, for example, the fact that process objects 11, 12 and 13 are assigned to the application object 10 is stored in the latter, that thread objects 14 and 15 are assigned to the process object 11 is stored in it, etc. It is furthermore stored in each object 11 to 21 by which object 10 to 17 it had been created, i.e. the information, that they had been created by the process object 11 is stored in the two thread objects 14 and 15. Exceptions are the application object 10, in which no information as to which object had created it is stored, and the module objects 18 to 21, to which no further objects are assigned.

After the entire software of the control has been created in a synchronized manner, an execution report is yet issued by the created objects 11 to 21 to the objects 10 to 17,

which had created them, that the creation was successful. This starts with the last created module objects 18 to 21 which, following a successful creation, transmit an execution report to the thread objects 14 to 17 which are higher-ranking than they. This execution report signals the successful creation of the module objects 18 to 21, which they had
5 created, to each thread object 14 to 17 and is only transmitted by the module objects if the creation of the module objects 18 to 21 really had been successful.

Once a thread object 14 to 17 has received an execution report regarding the successful creation of all module objects 18 to 21 assigned to it, and itself has also been created without errors, it also sends an execution report to the process object 11 to 13
10 which has created the thread object 14 to 17. This means that the thread objects 14 and 15 transmit the execution report to the process object 11 which already waits for it. As soon as a completed creation has been reported to the process object 11 by the thread objects 14 and 15, which had been created by it, the process object 11 also reports a completed creation to the application object 10. In the same way the thread object 17 reports a
15 completed creation to the process object 13 which thereafter reports a completed creation to the application object 10 in turn.

After the application object 10 has been informed by all process objects 11, 12 and 13, which it had created, that the structure was complete, the structure implemented by the application object 10 is available to the control. The first step of the synchronized start-up
20 of the control is therefore completed.

If an object cannot be created, so that an error exists, this object does not send a report that the creation had been successfully completed. Following a defined waiting period, the hierarchically higher-ranking object determines that the start-up could not have been successfully executed and also does not send an execution report to the object which
25 is higher-ranking. The corresponding function then is not available in the application 10. It is possible to determine at which location a creation had not been possible, based on execution reports which had not been received by the respectively higher-ranking objects.

Now the initialization of the communications channels 30, 31 between the objects 10 to 21 follows and proceeds in accordance with the same system as the creation of the
30 structure 10 to 21 of the software. In the process, at least logical communication channels 30, 31 are defined for the application object 10 and each process object, thread object and module object 11 to 21, by means of which they can communicate with already created

objects. In this case a communications channel 30, 31 can also be considered to be an object. For the creation of the communications channels 30, 31, information regarding the further objects with which messages are to be exchanged is already stored in the application object, the process object, thread object and module object 11 to 21.

5 Communications channels 30, 31 to these objects are then created.

After the structure 10 to 21 and the communications channels 30, 31 have been synchronously created, the synchronized initialization of the application takes place, for example with start values or adjustable parameters. This takes place in accordance with the same system as already described for the first step.

10 For ending the synchronized start-up, all objects 10 to 21 are informed of the transition to normal operations in accordance with this pattern. This also takes place in accordance with the same system already described above.

It is obvious to one skilled in the art that the distribution of the information to one or several files can take place arbitrarily. There is the option of storing the required
15 information on only a few files. For example, the start program can already contain extensive information regarding the entire structure to be created. Alternatively it is also possible to provide many small files, each of which contains only a small portion of the information required for initialization. Moreover, it is possible in connection with the synchronized start-up to pass through steps in addition to the creation of the structure, the
20 communications channels, the initialization and the transition to normal operations, in particular as a function of the machine tool to be controlled by the application 10.

The start-up described by way of example can be summed up as a whole as follows. In order to assure simple maintenance and extensibility, the programs for numerical control increasingly have an object-oriented structure. In this case the
25 applications 10, the processes 11 to 13, the threads 14 to 17 and the modules 18 to 21 are each represented by an object 10 to 21.

In the process, each object 10 to 21 sequentially assumes a number of defined states, such as object created, object initialized, start-up completed. Starting with the application object 10, first the processes 11 to 13, which are required for the application
30 10, are created, which hierarchically create further threads 14 to 17 necessary for the respective process 11 to 13, and the threads 14 to 17 in turn each create the required modules 18 to 21. In the process, each object 10 to 21 assumes a defined state as soon as

it itself has been completely created and when all objects 11 to 21 created by it have been completely created, and reports this to the creating object 10 to 17. When this confirmation arrives at the application object 10, it assumes the same state and triggers the next step of the start-up. In the process, each object 10 to 21 triggers the action in all objects 11 to 21 created by it and assumes the next defined state after it has completely performed the action, and when all objects 11 to 21 created by it have completely performed the action, and reports this to the creating object 10 to 17. All defined states are analogously passed sequentially in steps until normal operation has been achieved and the start-up is completed.

Advantageously the individual steps sequentially comprise the creation of the process 11 to 13, of the threads 14 to 17 and of the modules 18 to 21, until all required objects 11 to 21 have been created, thereafter the creation of the communications channels 30, 31 by the respective receiver, the opening of the communications channels by the transmitter, the parameterization of the modules 18 to 21 and the transition to normal operations of the numerical control.

APR 24 2001

example, in a drilling process a message regarding the number of revolutions of the drill is transmitted from the drill object to the object called spindle, furthermore messages regarding the position of the hole are transmitted to the objects of participating shafts, etc. Here, a standard interface is provided for the messages, so that they have a universal structure and can be designed independently of the involved objects. This standard interface for message exchanges between objects is implemented in connection with an object receiving or transmitting messages regarding the movement by means of a software kernel, which is intended to operate in real time and receives and transmits messages.

A method for a synchronized start-up of a numerical control is not disclosed there.

A system of several object manager components is described in DE 195 20 747 A1, on which object managers have been locally installed, which are to be initialized in several phases. Initialization takes place in several steps, wherein a new step is only started after the preceding step has been executed.

A method for a synchronized start-up of a numerical control with hierarchically arranged objects is not disclosed there.

It is therefore the object of the present invention to cite a method for a start of an application of a numerical control wherein their processes, threads and modules are embedded into a structure and are initialized in a synchronized manner, and wherein thereafter a transition to a normal operation is made. It is intended to assure that in the course of transitioning to normal operations the communications channels, by means of which modules, which are parts of different threads or processes, communicate with each other, are provided in particular.

This object is attained by means of a method having the features recited in claim 1.

Further developments and advantageous embodiments of the method in accordance with the invention can be taken from the dependent claims.

The method in accordance with the invention has the advantage that the software of a numerical control for a machine tool in the form of an application with the associated processes, threads and modules is started in a synchronized manner up to the

5
10

- 15
- 20

25

30

4.

further object (18 to 21) is hierarchically subordinated to each of the second objects (14 to 17), and

- the second objects (14 to 17) initiate the initialization step in the further objects (18 to 21), which are subordinated to them,

5 - after their own correct execution of the initialization step, the further objects
transmit an execution message to the second object (14 to 17), which is higher ranking
than they,

10 - after receiving the execution reports from all further objects (18 to 21) which are subordinated to them, and after their own correct execution of the initialization step, the second objects (14 to 17) transmit an execution report to the first object (11 to 13) which is higher ranking than they.

5. The method in accordance with claim 4, characterized in that the first objects are processes (11 to 13) of the application (10), and the second objects are threads (14 to 17), wherein the threads (14 to 17) make CPU time available.

6. The method in accordance with claim 5, characterized in that a definite function of the numeric control is implemented in each of the further objects (18 to 21).

20 7. The method in accordance with one of claims 1 to 6, characterized in that individual, essentially completed functions of the application (10) in the form of objects (10 to 21) are created in an initialization step.

8. The method in accordance with one of claims 1 to 7, characterized in that
25 communications channels (30, 31) are created between objects (10 to 21) in an
initialization step.

9. The method in accordance with one of claims 1 to 8, characterized in that parameters are assigned to the variables in the individual objects (10 to 21) in an initialization step.

10. The method in accordance with one of claims 1 to 9, characterized in that

the regular operation of the application (10) is released in an initialization step.

Abstract of the Disclosure

A method for the synchronized start-up of an application of a numerical control of a mechanized element that includes sequentially performing several initialization steps for several objects of an application, wherein several first objects are hierarchically subordinated to the application, and at least one second object is hierarchically subordinated to each of the several first objects, initializing the several first objects and the at least one second object, initializing the several first objects via the application and initializing the at least one second object via the first object. Transmitting a first execution report to one of the several first objects and transmitting a second execution report to the application via the first objects.

FIG. 1

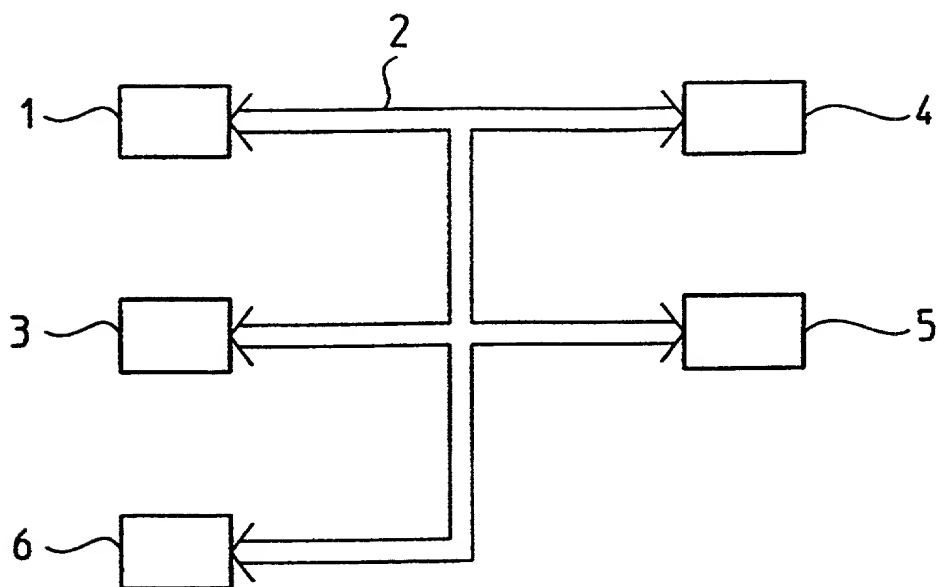
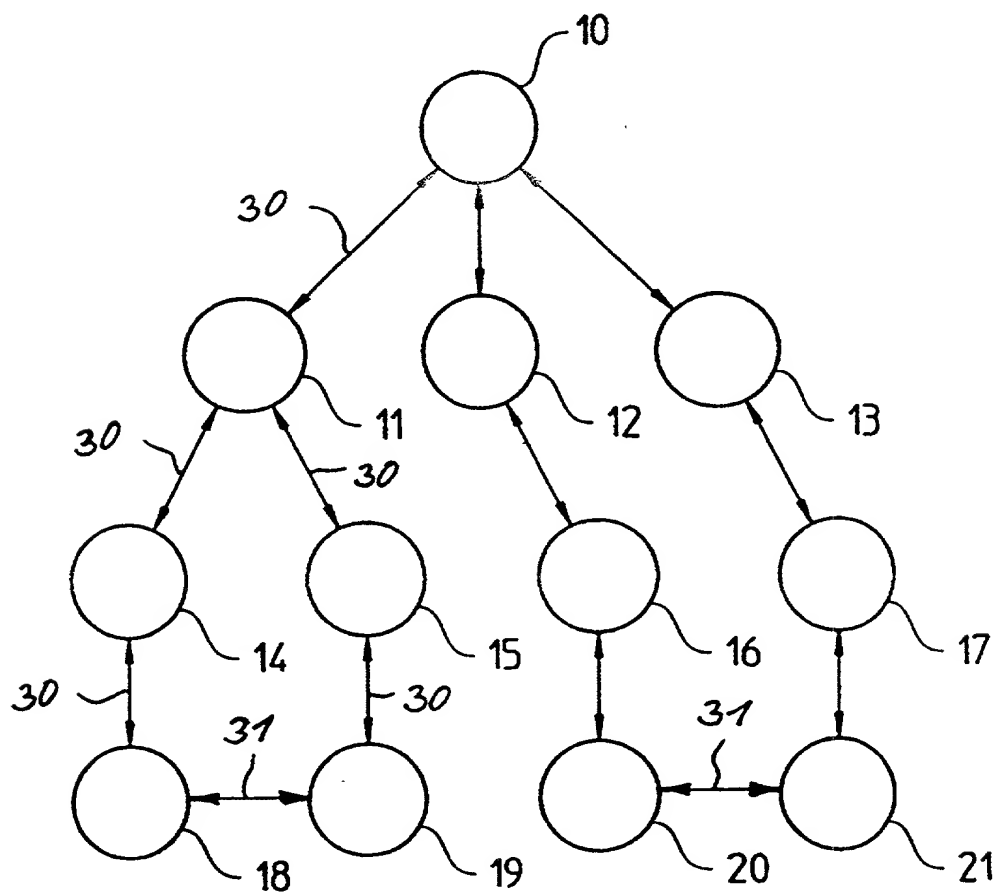


FIG. 2



Declaration and Power of Attorney For Patent Application Erklärung Für Patentanmeldungen Mit Vollmacht

German Language Declaration

Als nachstehend benannter Erfinder erkläre ich hiermit an
Eides Statt:

dass mein Wohnsitz, meine Postanschrift, und meine
Staatsangehörigkeit den im Nachstehenden nach meinem
Namen aufgeführten Angaben entsprechen.

dass ich, nach bestem Wissen der ursprüngliche, erste und
alleinige Erfinder (falls nachstehend nur ein Name
angegeben ist) oder ein ursprünglicher, erster und Miterfinder
(falls nachstehend mehrere Namen aufgeführt sind) des
Gegenstandes bin, für den dieser Antrag gestellt wird und für
den ein Patent beantragt wird für die Erfindung mit dem Titel:

METHOD FOR THE SYNCHRONIZED START-UP OF A
NUMERICAL CONTROL

deren Beschreibung

(zutreffendes ankreuzen)

☐ hier beigefügt ist.

☒ am November 28, 2001 unter der

Anmeldungsseriennummer 09/980,303

eingereicht wurde und am November 28, 2001
abgeändert wurde (falls tatsächlich abgeändert).

Ich bestätige hiermit, dass ich den Inhalt der obigen
Patentanmeldung einschliesslich der Ansprüche
durchgesehen und verstanden habe, die eventuell durch
einen Zusatzantrag wie oben erwähnt abgeändert wurde.

Ich erkenne meine Pflicht zur Offenbarung irgendweicher
Informationen, die für die Prüfung der vorliegenden
Anmeldung in Einklang mit Absatz 37, Bundesgesetzbuch,
Paragraph 1.56(a) von Wichtigkeit sind, an.

Ich beanspruche hiermit ausländische Prioritätsvorteile
gemäss Abschnitt 35 der Zivilprozessordnung der Vereinigten
Staaten, Paragraph 119 aller unten angegebenen
Auslandsanmeldungen für ein Patent oder eine
Erfindersurkunde, und habe auch alle Auslandsanmeldungen
für ein Patent oder eine Erfindersurkunde nachstehend
gekennzeichnet, die ein Anmeldedatum haben, das vor dem
Anmeldedatum der Anmeldung liegt, für die Priorität
beansprucht wird.

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as
stated below next to my name.

I believe I am the original, first and sole inventor (if only one
name is listed below) or an original, first and joint inventor (if
plural names are listed below) of the subject matter which is
claimed and for which a patent is sought on the invention
entitled

METHOD FOR THE SYNCHRONIZED START-UP OF A
NUMERICAL CONTROL

the specification of which

(check one)

☐ is attached hereto.

☒ was filed on November 28, 2001 as

Application Serial No. 09/980,303

and was amended on November 28, 2001
(if applicable)

I hereby state that I have reviewed and understand the
contents of the above identified specification, including the
claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is
material to the examination of this application in accordance
with Title 37, Code of Federal Regulations, § 1.56(a).

I hereby claim foreign priority benefits under Title 35, United
States Code, § 119 of any foreign application(s) for patent or
inventor's certificate listed below and have also identified
below any foreign application for patent or inventor's
certificate having a filing date before that of the application on
which priority is claimed:

German Language Declaration

Prior foreign applications
Priorität beansprucht

Priority Claimed

199 24 461.8	Germany	28/05/99
(Number)	(Country)	(Day Month/Year Filed)
(Nummer)	(Land)	(Tag; Monat; Jahr eingereicht)

<u>x</u>	
Yes	No
Ja	Nein

Ich beanspruche hiermit gemäss Absatz 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 120, den Vorzug aller unten aufgeführten Anmeldungen und falls der Gegenstand aus jedem Anspruch dieser Anmeldung nicht in einer früheren amerikanischen Patentanmeldung laut dem ersten Paragraphen des Absatzes 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 112 offenbart ist, erkenne ich gemäss Absatz 37, Bundesgesetzbuch, Paragraph 1.56(a) meine Pflicht zur Offenbarung von Informationen an, die zwischen dem Anmeldedatum der früheren Anmeldung und dem nationalen oder PCT internationalen Anmeldedatum dieser Anmeldung bekannt geworden sind.

I hereby claim the benefit under Title 35, United States Code § 120 of any United States application(s) listed below and insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, § 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

PCT/EP00/04856	27/05/00
(Application Serial No.)	(Filing Date)
(Anmeldesennummer)	(Anmeldedatum)

Completed	Completed
(Status)	(Status)
(patentert, anhängig, aufgegeben)	(patented, pending, abandoned)

Ich erkläre hiermit, dass alle von mir in der vorliegenden Erklärung gemachten Angaben nach meinem besten Wissen und Gewissen der vollen Wahrheit entsprechen, und dass ich diese eidesstattliche Erklärung in Kenntnis dessen abgebe, dass wissentlich und vorsätzlich falsche Angaben gemäss Paragraph 1001. Absatz 18 der Zivilprozessordnung der Vereinigten Staaten von Amerika mit Geldstrafe belegt und/oder Gefängnis bestraft werden koennen, und dass derartig wissentlich und vorsätzlich falsche Angaben die Gültigkeit der vorliegenden Patentanmeldung oder eines darauf erteilten Patentes gefährden können.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

German Language Declaration

VERTRETUNGSVOLLMACHT: Als benannter Erfinder beauftrage ich hiermit den nachstehend benannten Patentanwalt (oder die nachstehend benannten Patentanwälte) und/oder Patent-Agenten mit der Verfolgung der vorliegenden Patentanmeldung sowie mit der Abwicklung aller damit verbundenen Geschäfte vor dem Patent-und Warenzeichenamt:
(Name und Registrationsnummer anführen)

See Attached Exhibit "A"

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. (list name and registration number)

See Attached Exhibit "A"

John C. Freeman (312) 321-4262

John C. Freeman (312) 321-4262

Telefongespräche bitte richten an:
(Name und Telefonnummer)

Direct Telephone Calls to: (name and telephone number)

Postanschrift:
Brinks Hofer Gilson & Lione
P.O. Box 10395
Chicago, IL 60610

Send Correspondence to:
Brinks Hofer Gilson & Lione
P.O. Box 10395
Chicago, IL 60610

Voller Name des einzigen oder ursprünglichen Erfinders: Christian Rutkowski	Full name of sole or first inventor: Christian Rutkowski
Unterschrift des Erfinders Datum	Inventor's signature Date
Wohnsitz D-83278 Traunstein, Germany	Residence D-83278 Traunstein, Germany
Staatsangehörigkeit Germany	Citizenship Germany
Postanschrift Breslauer Ring 13 D-83278 Traunstein Germany	Post Office Address Breslauer Ring 13 D-83278 Traunstein Germany
Voller Name des zweiten Miterfinders (falls zutreffend) Thomas Klughammer	Full name of second joint inventor, if any Thomas Klughammer
Unterschrift des Erfinders Datum	Second Inventor's signature Date
Wohnsitz D-83339 Chieming, Germany	Residence D-83339 Chieming, Germany
Staatsangehörigkeit Germany	Citizenship Germany
Postanschrift Poststrasse 4a D-83339 Chieming Germany	Post Office Address Poststrasse 4a D-83339 Chieming Germany

Inventor(s): Christian Rutkowski and Thomas KlughammerTitle: METHOD FOR THE SYNCHRONIZED START-UP OF A NUMERICAL CONTROL**"Exhibit A"****POWER OF ATTORNEY**

The specification of the above-identified patent application:

☐ is attached hereto
☒ was filed on November 28, 2001 as application Serial No. 09/980,303

I hereby revoke all previously granted powers of attorney in the above-identified patent application and appoint the following attorneys to prosecute said patent application and to transact all business in the Patent and Trademark Office connected therewith:

2

John C. Freeman (34,483)
Kent E. Genin (37,834)

Please address all correspondence and telephone calls to John C. Freeman in care of:

Brinks Hofer Gilson & Lione
NBC Tower, Suite 3600
P.O. Box 10395
Chicago, IL 60610
(312)321-4200

The undersigned hereby authorizes the U.S. attorneys named herein to accept and follow instructions from _____ as to any action to be taken in the Patent and Trademark Office regarding this application without direct communication between the U.S. attorney and the undersigned. In the event of a change in the persons from whom instructions may be taken, the U.S. attorneys named herein will be so notified by the undersigned.

Christian Rutkowski
Inventor: Christian Rutkowski

Date: 3/8/2002

Thomas Klughammer
Inventor: Thomas Klughammer

Date: 3/8/2002